Especialización en Desarrollo de Software

ST1605 - Procesos Modernos de Desarrollo de Software

TAXONOMÍA DE METODOLOGÍAS DE DESARROLLO DE SOFTWARE

Introducción

Evolución del Proceso Software

El software se ha convertido en una de las tecnologías más importantes.

El software que se produce actualmente es la realización de la mayor parte de la propiedad intelectual.

Simplemente el mundo actual depende del software

Durante los últimos 40 o 50 años se han implementados varias metodologías de desarrollo de software para mitigar los nuevos riesgos y ayudar a los resultados planeados, se han desarrollado metodologías más estructuradas y controladas.

Evolución del Proceso Software

Lo que producimos no son bienes físicos sino ideas intangibles reflejadas en código binario, todos nuestros métodos tenían un enfoque principal: "**Gestión de requisitos de software**"

Requisitos de software fue la etiqueta que se aplicó a las abstracciones que se utilizan para llevar a la cadena de valor en el desarrollo y en la entrega a los clientes

Con el tiempo, las aplicaciones desarrolladas se volvieron aún más y más grandes y los métodos usados para controlar el trabajo se volvieron más y más pesados

Como una consecuencia no planeada, se comenzó a ir más lento en lo que se realmente intentaba ir más rápido: "la habilidad de entregar software de mayor valor y mayor calidad a más velocidad"

Evolución del Proceso Software

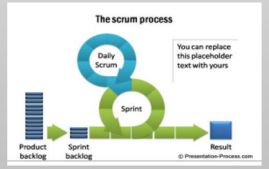
De tal forma que las dos décadas anteriores, el movimiento de las metodologías de desarrollo de software a más “ágiles” y “lean” ha sido uno de los factores que han afectado más significativamente la industria.

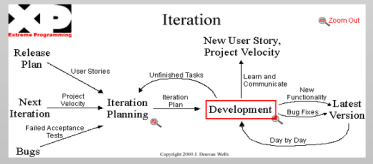
Simplemente se necesitan procesos que entreguen aún mejor seguridad y gobierno del que se ha experimentado pero sin ser una carga. Se quiere lo mejor de ambos mundos.

Evolución del Proceso Software

Procesos (Lean y Ágiles) Adaptativos a la Escala de la Empresa

Procesos (Ágiles) Adaptativos



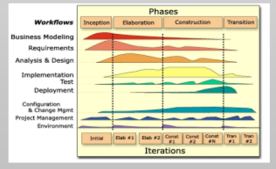


Crystal, SCRUM, XP, FDD, Lean, DSDM, Open UP, Kanban…

Procesos Iterativos

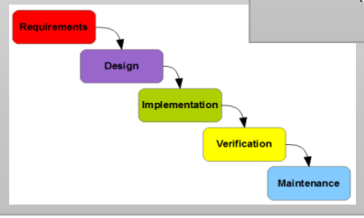
Spiral

RAD



UP

Procesos Predictivos



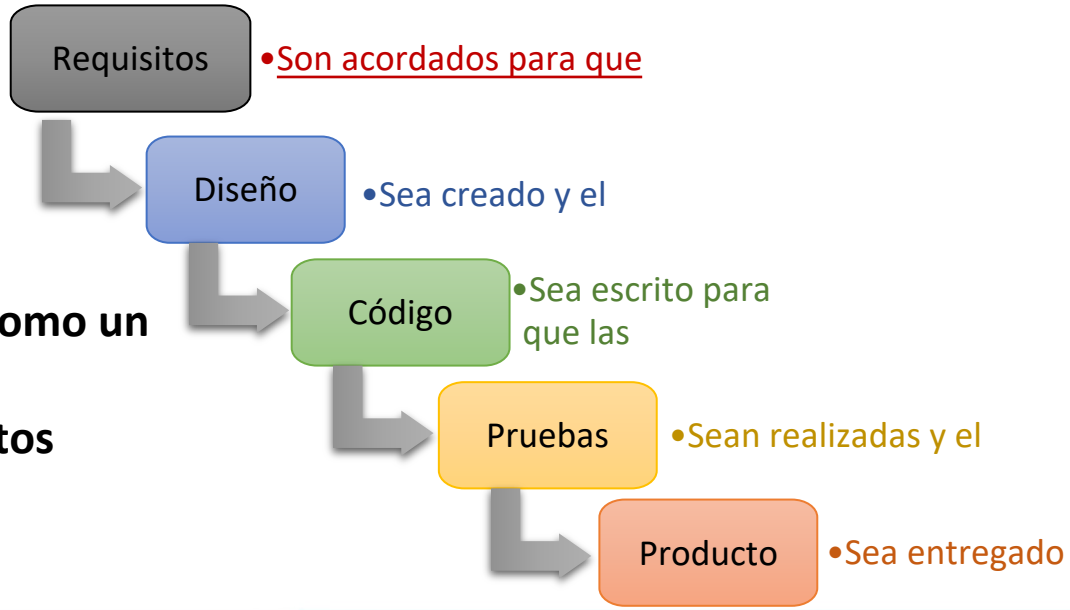
1970 1980 1990 2000 2010

Modelos de Proceso de Software - Predictivos

- *Procesos Predictivos (Cascada)*

- Aparece por la necesidad de predecir y controlar los resultados de los proyectos de software.

- Ocurre en un orden de etapas secuenciales:



- **Winston Royce su creador irónicamente lo describió como un modelo que no podría recomendarse para proyectos grandes.**

Modelos de Proceso de Software - Procesos iterativos e incrementales

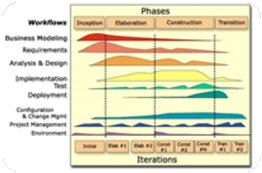
- Aparecieron en la década de los 80's con gran auge en los 90's, como respuesta a la necesidad de modelos *basados en descubrimiento*.
- Algunos de estos procesos:

Spiral

- Desarrollo rápido basado en el entendimiento del problema vía descubrimiento experimental.

RAD

- Rápida construcción de modelos, prototipos y sistemas iniciales usando herramientas avanzadas.



RUP

- Desarrollo iterativo e incremental de sistemas más grandes y complejos.

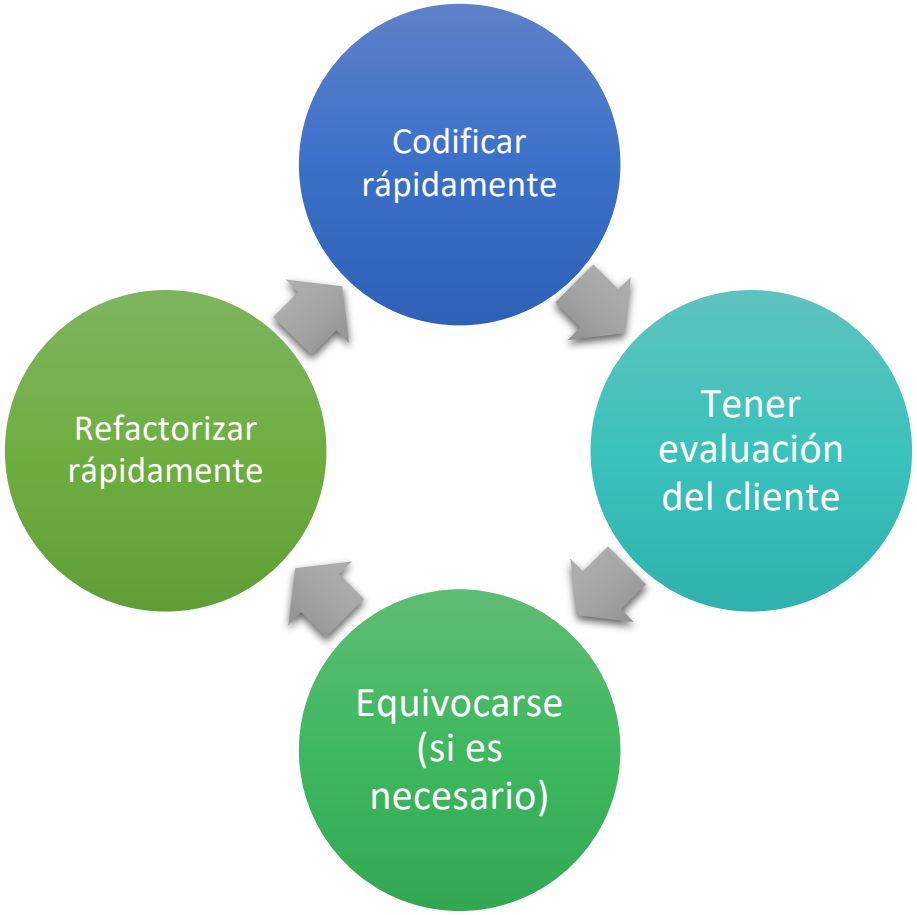
Modelos de Proceso de Software - Procesos Adaptativos

- Ágiles

- A finales de los 90's, comenzó un movimiento por modelos más livianos y adaptativos, los cuales han avanzado y se han mantenido hasta nuestros días.

- Se basa en algunos paradigmas fundamentales:
  - Orientación a objetos
  - Lenguajes de 3ra generación
  - TDD (Test Driven Development)

- Este modelo asume que con las herramientas y prácticas adecuadas, es más costo efectivo



Es mejor que tratarse de anticipar y documentar todos los requisitos por anticipado

Proceso Software

Concepto de Ingeniería Software (SWEBOK)

La IEEE Computer Society define la ingeniería de software como:

(1) La aplicación de un enfoque disciplinado cuantificable y sistemático, para el desarrollo, operación y mantenimiento de software, es decir, la aplicación de ingeniería al software.

(2) El estudio de los enfoques como en (1).

Concepto de Ingeniería Software (Yingxu Wang)

La Ingeniería de software es una disciplina de la ingeniería que estudia la naturaleza del software, enfoques y metodologías de desarrollo de software a gran escala, las teorías y las leyes detrás de las conductas de software y prácticas de ingeniería de software.

La naturaleza de la ingeniería de software, sus teorías y metodologías están determinadas por la naturaleza de los objetos de estudio, software y las necesidades de medios matemáticos, teóricos, metodológicos adecuados y denotativos de esta disciplina.

Aunque la ingeniería de software se ha acumulado un rico conjunto de principios empíricos y heurísticos, algunos de ellos se han perfeccionado y formalizado con el fin de formar teorías coherentes para la ingeniería de software.

Se reconoce que la ingeniería de software requiere la investigación teórica y empírica. El primero se centra en las fundaciones y las teorías básicas de la ingeniería de software, mientras que la segunda se centra en las herramientas / ambientes y mejores prácticas

Concepto de Proceso Software (1)

Proceso Software

- Es un conjunto de prácticas secuenciales que son funcionalmente coherentes y reutilizables para organizar, implementar y administrar proyectos de Software.
- Un conjunto coherente de políticas, estructuras organizacionales, tecnologías, procedimientos y artefactos que son necesarios para concebir, desarrollar, instalar y mantener un producto software (Fugetta, 2000)

Categoría de Procesos

- Un conjunto de procesos que son funcionalmente coherentes y reutilizables en un aspecto de ingeniería de Software.

Concepto de Proceso Software (2)

Sistema de Procesos

- Es un **conjunto completo de procesos** de software estructurados, descrito por un modelo de procesos.

Modelo de Procesos

- Un modelo de procesos es un plan para organizar, implementar, guiar y administrar procesos de ingeniería de software en una organización con un sistema de procesos y **buenas prácticas** establecidas y validadas (Wang, King,2000).

Práctica

- Es una actividad o un estado en un proceso de ingeniería de software que conlleva a una tarea específica de los procesos.
- Una práctica es **la mínima unidad** que puede ser modelada en un sistema de procesos.

Modelos de Procesos

Modelo de un sistema de proceso software (MPSS)

De acuerdo a la teoría de sistemas, un sistema de procesos de Ingeniería de software es un sistema dinámico, discreto, distribuido y no determinístico.

Un MPSS puede ser **empírico** o **formal** desde el punto de vista de las técnicas de modelamiento o puede ser **descriptivo** o **prescriptivo** desde el punto de vista del propósito del modelamiento.

Empírico / Formal

Modelo de procesos empírico

- Es un modelo que define un sistema de procesos software organizado y comparable con buenas prácticas implementadas en la industria del software.

Modelo de procesos formal

- Es un modelo que describe la estructura y la metodología de un sistema de proceso software con un enfoque algorítmico o por un lenguaje descriptivo de abstracción de procesos.

Descriptivo / Prescriptivo

Modelo de procesos descriptivo

- Es un modelo que describe "**qué hacer**" acorde a determinado sistema de proceso software.

Modelo de procesos prescriptivo

- Es un modelo que describe "**cómo hacerlo**" acorde a determinado sistema de proceso software.

Modelos Empíricos / Descriptivos

Los modelos empíricos y descriptivos son los más comunes, son diseñados para contribuir a responder preguntas frecuentes como:

- ¿Qué son buenas prácticas en ingeniería de software?
- ¿Qué son experiencias exitosas en una organización de desarrollo de software en sus procesos?
- ¿Cómo es un sistema de procesos de ingeniería de software establecido?
- ¿Cómo son los estados y el rendimiento de un sistema de procesos implementado y controlado?
- ¿Cómo es el mejoramiento de un sistema de procesos de ingeniería de software existente?

Estructura de un Modelo de PS

Establecimiento de sistema de procesos software (SPE)	• Es un procedimiento sistemático para seleccionar e implementar un sistema de procesos basados en un modelo a la medida, extensión y/o técnicas de adaptación.
Evaluación de sistema de procesos software (SPA)	• Es un procedimiento sistemático para investigar la existencia, adecuación y rendimiento de un sistema de procesos implementado a partir de un modelo, estándar o punto de referencia.
Mejora de sistema de procesos software (SPI)	• Es un procedimiento sistemático para mejorar el rendimiento de un sistema de procesos existente por medio de cambios a los procesos o actualización de nuevos procesos con el fin de corregir o abolir problemas identificados.

Naturaleza del Proceso Software (1)

Es complejo	• La dependencia de dominios y de otras tecnologías implican diferentes variaciones del mismo proceso.
No es un proceso de producción	• Dirigido por excepciones • Muy determinados por circunstancia impredecibles • Cada instancia de aplicación del proceso tiene sus peculiaridades
No es un proceso de ingeniería “pura”	• Alta dependencia de la gente • El diseño y la producción no están claramente separados • Es difícil la planificación de presupuestos, calendarios y calidad

Naturaleza del Proceso Software

Está basado en descubrimientos que dependen de la comunicación, coordinación y cooperación dentro de marcos de trabajo predefinidos

- Los entregables generan nuevos requerimientos
- Los costos de cambio del software no suelen reconocerse
- El éxito depende de la participación coordinada y activa de todos los involucrados (usuario, cliente, sponsor, operación, equipo de desarrollo etc.)

Son procesos basados en el conocimiento

Cuerpo de Conocimiento Ingeniería de Software - SWEBOK

El cuerpo de conocimientos de ingeniería de software es un término que describe toda la suma de conocimiento dentro de la profesión de la ingeniería de software.

La guía del Cuerpo de Conocimiento de la Ingeniería de Software busca identificar y describir ese subconjunto del conjunto de conocimientos que se acepta en general o, en otras palabras, el cuerpo básico de conocimientos.

El conocimiento generalmente aceptado está relacionado con otros tipos de conocimiento.

En SWEBOK se proponen tres categorías de clasificación de los conocimientos.

Conocimientos Generalmente Aceptados:
Prácticas tradicionales bien establecidas y recomendadas por la mayoría de las organizaciones

Conocimientos Avanzados y de Investigación:
Prácticas innovadoras que están siendo probadas y/o utilizadas por pocas organizaciones y conceptos que todavía están siendo desarrollados y probados por organizaciones de investigación.

Conocimientos Especializados:
Prácticas que aplican sólo para cierto tipo de aplicaciones

Guía del SWEBOK

La guía de Conocimiento de la Ingeniería de Software (SWEBOK) se estableció con los siguientes cinco objetivos:

1. **1.** Promover una visión consistente de Ingeniería de software en todo el mundo
2. **2.** Aclarar el lugar y establecer el límite de ingeniería de software con respecto a la otra disciplinas como la informática, la gestión de proyectos, la ingeniería informática, y las matemáticas
3. **3.** Caracterizar el contenido de la disciplina de ingeniería software
4. **4.** Proporcionar un acceso a los tópicos en el Software Engineering Body of Knowledge
5. **5.** Sentar las bases para el desarrollo del currículo así como de certificación y concesión de licencias individual

Áreas de Conocimiento SWEBOK 2004 V3.0

Software requirements	Software design	Software construction	Software testing
Software maintenance	Software configuration management	Software engineering management	Software engineering process
Software engineering Models and methods	Software quality	Software engineering Professional Practice	Software engineering Economics

Computer Fundations	Mathematical Foundations	Engineering Fundations

Concepto de Metodología

Metodología

General

- Alternativamente puede definirse la *metodología* como el estudio o elección de un método pertinente para un determinado objetivo

En Software

- Conjunto de procedimientos, técnicas, herramientas y un soporte documental que ayuda a los desarrolladores a realizar nuevo software.

Marco de Trabajo

Marco de trabajo ("framework")

Define, en términos generales, un conjunto estandarizado de conceptos, prácticas y criterios para enfocar un tipo de problemática particular que sirve como referencia, para enfrentar y resolver nuevos problemas de índole similar.

Componentes de un Marco de Trabajo

Tienen definidos comúnmente prácticas orientadas a:

- Planeación
- Comunicación
- Artefactos que se generan
- Prácticas relacionadas con: Modelos, Construcción, Despliegue
- Roles

Componentes de una Metodología

Es específico y prescribe:

- Fases
- Disciplinas con sus Actividades: de Ingeniería de Software y de Gestión General
- Artefactos a Generar
- Roles
- Métodos y Técnicas
- Herramientas

Aproximación a la Taxonomía Metodologías

Importancia de la clasificación

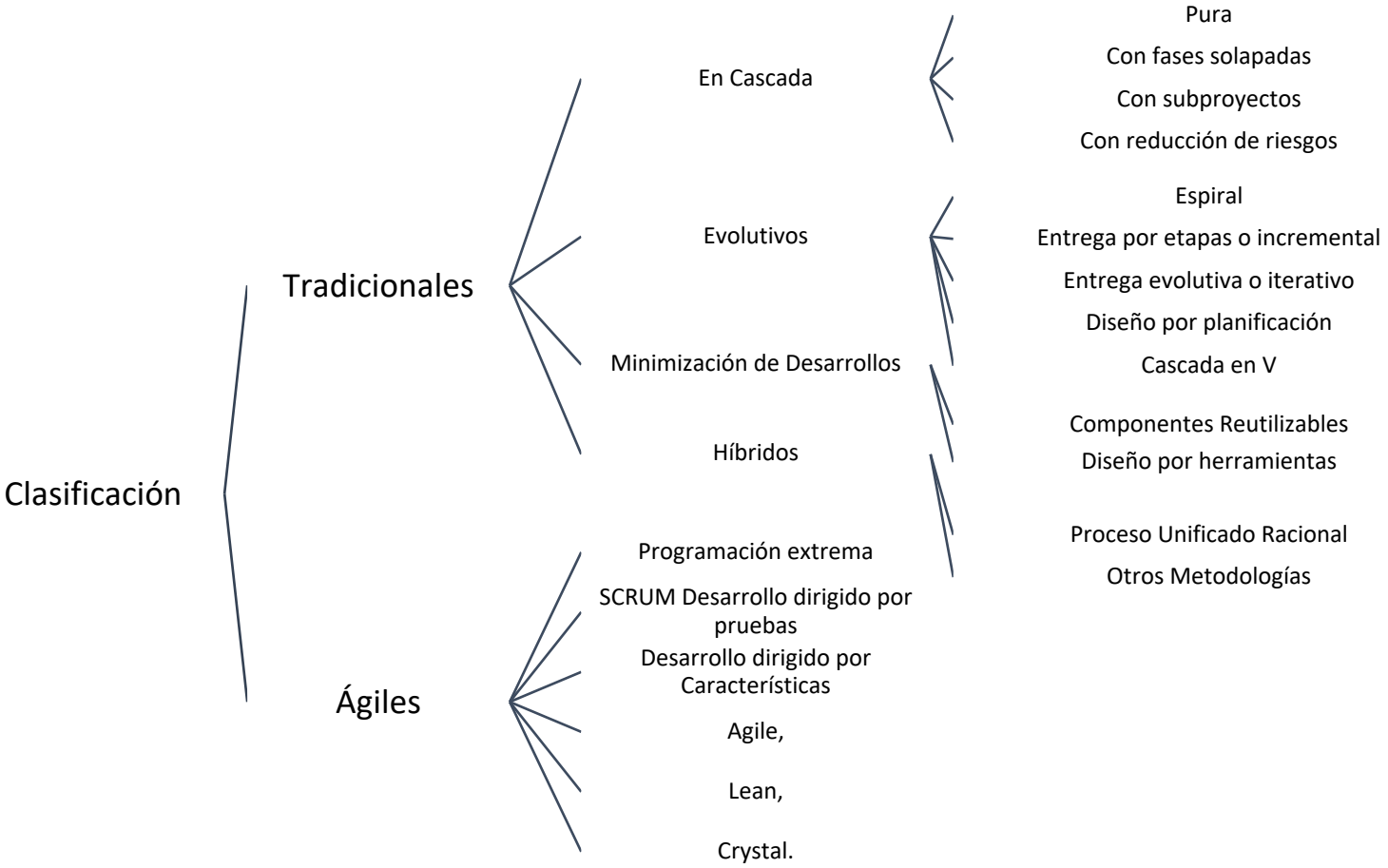
En ciencia e ingeniería, una descripción sistemática y la organización de los sujetos investigados ayudan a mejorar el conocimiento en este campo.

Esta organización se puede lograr a través de la clasificación de los conocimientos existentes.

La clasificación del conocimiento ha apoyado la maduración de diferentes campos de conocimiento principalmente de cuatro maneras:

- La clasificación de los objetos de un campo de conocimiento proporciona una terminología común, que facilita el intercambio de conocimientos.
- La clasificación puede proporcionar una mejor comprensión de las interrelaciones entre los objetos de un campo de conocimiento.
- La clasificación puede ayudar a identificar brechas en un campo de conocimiento [1–3].
- La clasificación puede respaldar los procesos de toma de decisiones.

Aproximación Empírica



Aproximación más formal

Se enfoca en clasificar recomendaciones expresadas en modelos de proceso. Estas recomendaciones se han agrupado en seis ejes principales, a saber,

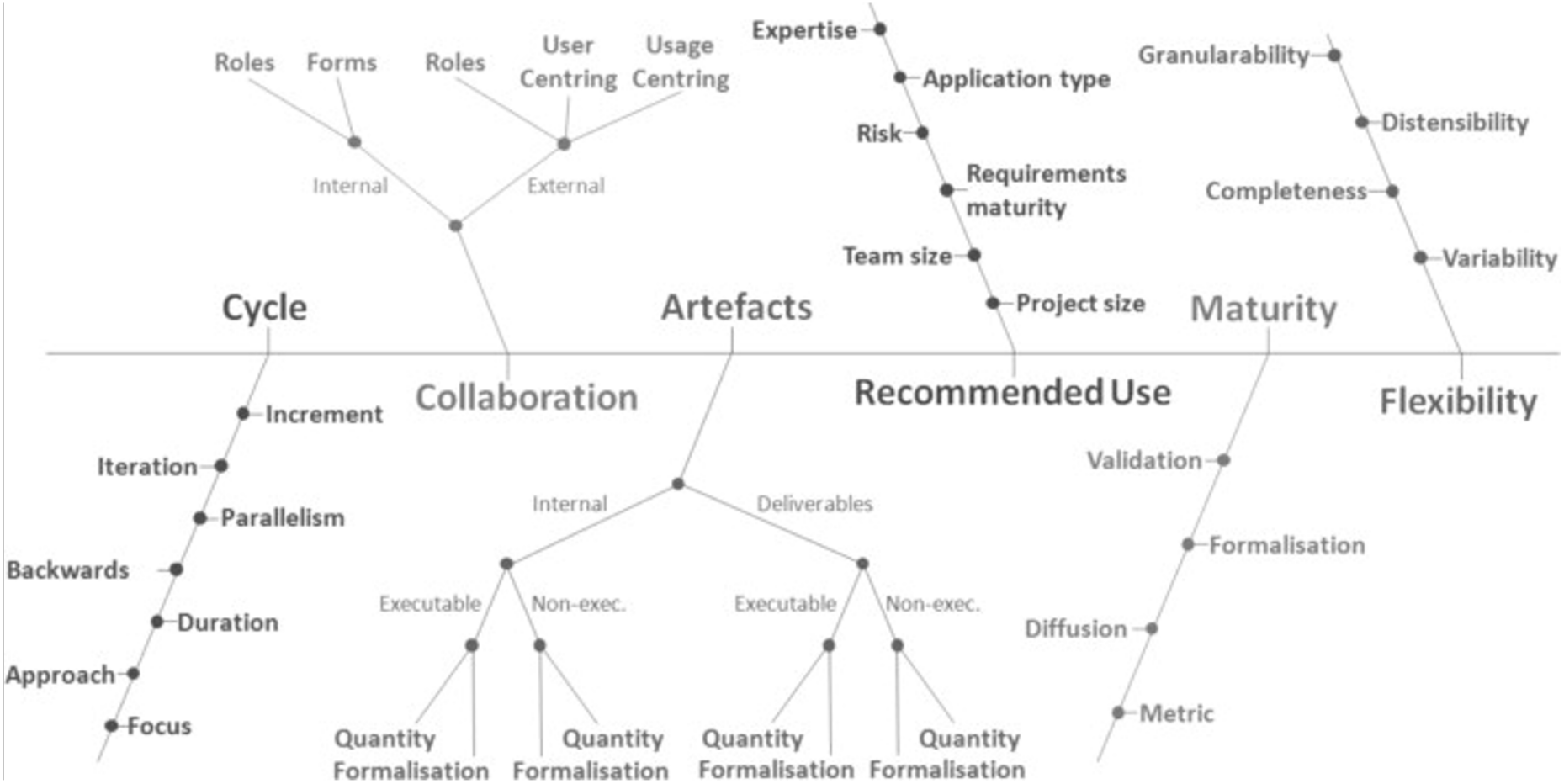
- Ciclo
- Colaboración,
- Artefactos,
- Uso recomendado,
- Madurez,
- Flexibilidad

Clasificación

El ciclo	• Se refiere a las recomendaciones de métodos con respecto a la granularidad de iteración, la duración del ciclo de vida, los incrementos, la gestión de contragolpes, el enfoque de arriba hacia abajo o de abajo hacia arriba.
La colaboración	• Categoriza los tipos de relación que pueden existir entre los miembros del equipo de diseño / desarrollo y entre ellos y los demás interesados; Enumera las características de los enfoques centrados en el usuario y en el uso.
Los artefactos	• Clasifican los tipos de prototipos y documentos a producir (interno / entregable, ejecutable vs. no ejecutable, formalizado o no).

Clasificación

El método de uso recomendado	• Identifica las características inherentes al contexto del método, como el tipo (s) de proyecto, el tamaño del equipo, el conocimiento requerido, la madurez de los requisitos, las situaciones administradas (plasticidad de interfaz de usuario, multinúcleo, computación en la nube).
La madurez	• Mide la evolución de los métodos que resultan de su uso real.
La flexibilidad	• Mide la adaptabilidad del modelo de proceso; por ejemplo, explica posibles elecciones o extensiones.



Tomado de: Eric Céret, Sophie Dupuy-Chessa, Gaëlle Calvary, Agnès Front, Dominique Rieu, A taxonomy of design methods process models,

Referencias Bibliográficas

Bibliografía y Referencias

Wang, Yingxu & King, Graham (2000). Software Engineering Processes. Principles and Applications. CRC Press LLC, N. W. Florida.

Tong Li. An Approach to Modelling Software Evolution Processes. Springer 2008.

Wang, Yingxu. SOFTWARE ENGINEERING FOUNDATIONS A SOFTWARE SCIENCE PERSPECTIVE. Auerbach Publications 2008.

Bibliografía y Referencias

Alexander, Ian; Stevens, Richard. Writing better requirements. Pearson Education Ltd., Great Britain © 2002

Young, Ralph R., The Requirements Engineering Handbook. ARTECH HOUSE, INC., Boston, © 2004

Leffingwell, Dean., Agile Software Requirements Lean Requirements Practices for teams, Programs, and the enterprise, Pearson Education, Inc., Boston, © 2011

SWEBOK V3. Software Engineering Body of Knowledge. IEEE Computer Society.

Bibliografía y Referencias

Hugh Dubberly. How Do You Design Process. Cap: Software development models Pag. 67-81.

Eric Céret, Sophie Dupuy-Chessa, Gaëlle Calvary, Agnès Front, Dominique Rieu. A taxonomy of design methods process models, Information and Software Technology, Volume 55, Issue 5, 2013, Pages 795-821, ISSN 0950-5849, https://doi.org/10.1016/j.infsof.2012.11.002.

Frank Bomarius, Markku Oivo, Päivi Jaring and Pekka Abrahamsson Editors. Product-Focused Software Process Improvement.

Raydmond J. Madachy. Software Process Dynamics. Wiley-IEEE Press.

Discusión para la práctica

- Conformemos equipos de 3 y realicemos un análisis riguroso:
  - Cuáles son las prácticas de desarrollo de software que más valor genera para sus empresas.
  - Realice un análisis general de cómo esas prácticas contribuyen a generar valor para las compañías que los desarrolla. Clasifique (muy alto, alto, medio, bajo, muy bajo.
  - Liste de acuerdo al conocimiento de todos las prácticas que consideran deberían existir.

Fin de Tema

Procesos Modernos de Desarrollo de Software

TAXONOMÍA DE METODOLOGÍAS DE DESARROLLO DE SOFTWARE